DESCRIPTION

Device for increasing the performance capability of processor systems

The present invention relates to a device for increasing the performance capability of processor systems according to the preamble of Claim 1.

Large servers comprise a processor system having a large number of individual processors. The number of individual processors is, for example, 16, 32, 64, 128, etc. Such servers are known as UNIX or as NT systems. The individual processors currently usually already have a cache memory in the chip. Therefore, an on-chip first level cache memory or a first level cache memory is also referred to. The individual processors may additionally be assigned still further cache memories outside the chip. A first further such cache memory is then referred to as an off-chip second level cache memory or a second level cache memory. In addition to the first and second level cache memories, additional cache memories may be provided, which are then referred to as third level cache memories, for example.

The additional cache memories are used in particular to assemble individual processors into a cluster each individually or in groups. Current standard processors only allow the construction of up to four individual processors in each case. If multiple such additional cache memories having four individual processors each, for example, are provided, the above-mentioned large processor systems may be constructed. The individual additional cache memories must be coupled to one another so that the cache is coherent for this purpose. The coupling may contain hierarchy stages, in which not only four individual processors, but rather even multiple additional cache memories are coupled to one another in each case. As a result, a processor system having a large number of individual processors is obtained, which has a large performance capacity.

As noted above, a processor system must behave cache-coherently. For this purpose, it is possible to observe and interpret the bus transactions. From the analysis of the bus transactions, it may be concluded in which state situations the cache data blocks of the cache memories affected by the bus transactions are located. If it is known in which state situations the cache data blocks are located, it may be ensured that cache coherency is produced and/or maintained within the processor system.

The states of the cache data blocks may be described using a cache protocol. A known cache protocol follows the MESI standard. According to the MESI standard, the individual cache data blocks are assigned to one of four predefined MESI states. The four predefined MESI states are: MODIFIED (M), EXCLUSIVE (E), SHARED (S), and INVALID (I). These states may be represented in code by two MESI bits.

MODIFIED means that the associated cache memory block is contained exclusively in the cache memory, but has been newly written. It is currently not known in the remaining processor system. EXCLUSIVE means that the associated cache memory block is not changed. It corresponds to the content of the system main memory. SHARED means that the associated cache memory block is still located in a further cache memory and is still valid. INVALID means that the associated cache memory block is invalid or is not present in the TAG RAM of the memory.

The state of a cache memory block may be altered by a particular assigned individual processor via a read and write procedure. The state may also be altered by an internal system query, also known as snooping. It may also be altered by external logic units, e.g., another individual processor or a second level cache memory, i.e., through external snooping. In each case, a particular cache memory block is only in one of the four MESI states at each instant.

A more precise functional description of the MESI protocol is disclosed, for example, in the Internet at the Internet address: http://www.altavista.com/egi-bin/query?pg=q&kl=de&q=MESI&search=Search, point 1, "Pentium: Neuerungen in der x86 Architektur [Innovations in the x86 Architecture]", last changed on 03/28/97 and/or in the associated publication in the Internet having the Internet address: http://plweb.htu.tuwien.ac.at/pentium, point 2.7.1 M.E.S.I.-Protokoll [MESI protocol].

A possibility for ensuring the cache coherency in a processor system having additional cache memories is for the additional cache memories to have a superset of all cache data blocks of the other cache memories to which it is directly connected. For this purpose, it is necessary that during all displacement procedures, an update request is directed, for example, to a connected individual processor and/or its cache memory and a cache memory block affected by the displacement procedure is requested back. For update requests of this type and/or their processing, system performance is required, which is not available to a user as usable performance. The performance capability of the overall system is thus restricted.

The object of the present invention is therefore to specify technical measures, through which the performance capability of a large processor system is increased in particular.

This object is achieved according to the present invention by a device in the processor system which has the features of Claim 1.

Accordingly, the additional cache memories not only have simple cache memory block management implemented via the states M, E, S, and I, but rather combined state management between each additional cache memory and/or the cache memories of the individual processors is implemented. This combined state management is shown in the expansion of the MESI states of the cache protocol of the additional cache memories. The advantage is that an increased number of update requests in the direction of other points in the processor system may be saved, since on the basis of the expanded MESI states, it may be decided beforehand that affected requested cache data blocks could not be at the other points in the processor system. The performance outlay which was required until now for these update requests is therefore now available to the processor system as usable performance. The performance capability of the processor system is thus increased.

In addition, the circumstance that because of the expanded MESI states, some cases may again be decided, in which handling is necessary, but which may be fulfilled more rapidly overall, also contributes to an increase of the performance capability of the processor system. For example, there are cases in which, because of a preceding update request from a part of the processor system to an additional cache memory, a following update request performed by the additional cache memory to lower cache hierarchies is necessary. For example, so that an affected cache memory block is assigned the MESI state I. In the cases mentioned, because of the present expanded MESI state, it may be known that modified data is still to be transmitted previously by the affected cache memory block. In these cases, for example, the additional cache memory may immediately communicate to the part of the processor system initiating the preceding update request that the noted MESI assignment has occurred, although this assignment is actually caused only in a subsequent measure by the additional cache memory. The processor system therefore does not need to wait until the confirmation of the actual end of the update request, but rather may begin immediately with the next tasks.

Advantageous embodiments of the present invention are the subject matter of subclaims.

Accordingly, a processor system is structured hierarchically in regard to its cache memory structure, in order to obtain a maximum of saved update requests in the direction of lower-lying hierarchy levels by collecting update requests already at the highest possible hierarchy levels.

By selecting suitable MESI states, the number of required MESI states may be kept as low as possible. At the lowest possible number of MESI states, the lowest possible number of coding points is required in the digital coding of the MESI states. The hardware outlay is thus reduced.

In the following, the present invention is explained in greater detail on the basis of a drawing.

Figure 1	shows a schematic circuit of a large processor system according to the related art,
Figure 2	shows a state transition diagram of MESI states according to the related art,
Figure 3	shows a state transition diagram of expanded MESI states according to the present invention, and
Figure 4	shows a table having situations in which system performance is saved in relation to a use of MESI states as shown in Figure 2 and may be used as

usable performance of a processor system according to Figure 1 if expanded MESI states as shown in Figure 3 are used.

Figures 1 through 4 are to help explain the state of affairs according to the present invention. They do not make any claim to completeness. The following description thus relates to a possible embodiment of the related art and of the present invention.

Figure 1 shows two groups having individual processors EP, which are each connected via first buses BS1 to an additional cache memory CS. The particular additional cache memories CS are connected via a joint second bus BS2 to a main memory component MEM. The second bus BS2 may also be identified as a cache or system bus. Besides the additional cache memories, all types of I/O or connection systems to further system components may be connected to the second bus BS2. In the present exemplary embodiment, no further systems are at least shown connected in the drawing. As the second bus BS2 may be identified as a cache or system bus, the first bus BS1 may be identified as a processor bus.

The processor system shown in Figure 1 is constructed hierarchically. From the view of the additional cache memory CS, lower hierarchy stages are situated in the direction of first buses BS1 and higher hierarchy stages are situated in the direction of second bus BS2.

The connections between the individual processors EP and the particular associated first bus BS1, between the first buses BS1 and the particular additional cache memories CS, between the additional cache memory CS and the second bus BS2, as well as between the second bus BS2 and the main memory component MEM, are each implemented as bidirectional, so that data and codes and other required information may be transmitted between all components in all directions.

The individual processors EP each have an internal cache memory and an external cache memory, which are not shown in greater detail in Figure 1. The cache hierarchy which is formed by the two cache memories of an individual processor EP together is identified in the following as the second-level cache memory (SLC). When one refers to the SLC, it is also to be understood as the associated individual processor EP which controls the two cache memories. Vice versa, when one addresses an individual processor EP, the two internal and external cache memories belonging to the individual processor EP are addressed.

The short form SLC is used in Figures 2 through 4. In relation thereto, the additional cache memories CS are identified as the third level cache memory (TLC). The short form TLC is also used in Figures 2 through 4. The short form TLC is used alternately in the following instead of the term "additional cache memory CS".

Requests between the additional cache memories CS and the individual processors EP are to be understood as internal requests, while requests between the additional cache memories CS and the main memory component MEM and/or the other components coupled to the cache memory bus BS2, such as I/O components, are to be understood as external requests.

Commands which initiate the requests and their processing in the processor system, and play a role in maintaining the cache coherency may be, for example: PREFETCH, READSHARED (RDS), READEXCLUSIVE (RDE), READMODIFIED (RDM), and WRITE (WR). These commands cause transitions of states which are assigned to the affected cache data blocks on the basis of a cache protocol operating according to the MESI principle to maintain the cache coherency. Possible state transition diagrams may be seen in Figures 2 and 3.

The state transition diagram shown in Figure 2 is based on the fact that the above-mentioned cache protocol operates using four MESI states. The state transition diagram shown in Figure 3 is based on the fact that the above-mentioned cache protocol operates using expanded MESI states. In the present case, the cache protocol operates using 8 MESI states. The state transition diagrams of Figures 2 and 3 are to be seen from the viewpoint of a memory block of an additional cache memory CS. The cache memories of the individual processors EP have corresponding state transition diagrams.

The following description relates to Figure 2.

Starting from a state I of an affected cache memory block of an additional cache memory CS, which indicates that the entry of the affected cache memory block is invalid, an internal RDS command directed to the additional cache memory CS in regard to this cache memory block finally results in a state transition from the state I to the state S. Previously, however, the entry was updated by the additional cache memory CS since the entry existing in the additional cache memory CS was invalid. For this purpose, queries at the other cache memories are required in order to establish where a more current entry is to be found. This procedure is also referred to the following as a TLC command. After completion of the TLC command, the entry is a jointly used, valid entry in the additional cache memory CS. This is true correspondingly when, with the same starting position, instead of the RDS command a prefetch request is performed, under the condition that such a possibility is provided at all.

In the boxes below the state transition diagram, a list of the possible four states is shown in the left box. The states possible in the particular cases are shown next to it for the SLC. The states possible for the TLC are also shown. Since the observation proceeds from a TLC, the states correspond to the states specified in the first column of the box. Although it is not always explicitly specified in the following for the sake of simplicity, the statement of MESI states relates to particular cache memory blocks in an affected cache memory.

The first line of the box under discussion shows that the state I also exists in the SLC in the event of a state I in the TLC. This is because, in the present exemplary embodiment of the TLC, a superset does not absolutely necessarily have the actual entries, but does have the information about the states of the entries of the connected SLC. This has the result that only equal and/or lower-value states always occur in the SLC. Vice versa, the state in the TLC is always equal or higher-value than the state of the SLC.

The state Exclusive forms an exception, since the state is essentially only advisable when there is at least the intention of modifying the affected entry. Since the TLC does not know in the event of an Exclusive request whether the requested entry is actually modified, it nonetheless has to assume that the entry is modified. In the event of an Exclusive request, the actual state in an SLC may thus also be M. This circumstance plays no role, because a Modify request is finally an Exclusive request, solely with the certainty that the entry is modified.

If an entry in the TLC is already marked as S and the TLC receives an RDS command in regard to this entry, the MESI state of the entry does not change. However, the entry in an SLC may be discarded again after it has been retrieved from the TLC. Classification of the MESI states in regard to this entry in the SLC by the associated individual processor EP is thus possible. This does not play a role for the TLC, since the entry itself was not changed and thus still has validity overall. The TLC would note a change because the entry would have had to have been previously brought into a E and/or M state beforehand by the affected individual processor. Authorization of a modification is not connected to the RDS or prefetch command. One of the MESI states I or S for an affected entry may thus actually be behind the state S of an entry in the TLC in regard to this entry in the SLC, i.e., in general in lower cache hierarchies. This state of affairs is indicated in the above-mentioned lower left box in the second line.

This principle also applies for the following observations, because of which it will not always be discussed in such detail therein.

The commands RDE or RDM request an entry Exclusive from the TLC. Independently of whether the state I or S has previously been marked for the entry in the TLC, the entry is subsequently marked as the state E in the TLC. According to the statements above, one of the states I, S, E, or M may finally be actually assigned for this entry in the SLC. This state of affairs is specified in the lower left box in the third line.

Each of the commands RDE or RDM causes a TLC command, i.e., a bus request, since the TLC must ensure before the transfer of the entry that starting from the affected TLC in the lower cache hierarchies of the processor system, this entry is marked with the state I.

If an SLC has requested an entry of the TLC using the command RDE or RDM and then modified it, it has to ensure that the modified entry is written back in the TLC using the WR command. The TLC subsequently marks the modified entry with the state M.

After the modification of the requested entry, the requesting SLC may maintain the entry in modified form and identify it as such. It may also write back the entry and, as a result of writing back, downgrade the state of the entry and only still put it in the state E, for example. If the entry is to be requested or suppressed in the SLC, the entry may correspondingly be assigned either the state S or the state I by the SLC. This is possible because the state M is of the highest value, before the states E, S, or I in this order. The preceding state of affairs is specified in the box shown on the bottom left in Figure 2 in the fourth line.

If an RDS, RDE, or WR command is received at the TLC in regard to the affected cache memory block having the state M, the state for this entry in the TLC does not change. If an RDM command goes to the TLC, in an advantageous embodiment of the present invention, the notation M may be given to the state for the affected entry instead of the notation E. This has the advantage that in many cases the cache memory block will not be read unnecessarily. If the cache memory block is requested by an individual processor EP situated in a lower cache hierarchy using RDM, the TLC knows that the cache memory block is actually changed. This information is part of the RDM request. If the cache memory block was not definitely changed, it would be requested using the RDE command. In other words, if a cache memory block is requested using an RDM command, the requesting individual processor and/or the associated SLC finally has the newest data Exclusive. If this state of affairs is also noted as such, the data is not first sought in the TLC in the event of an access to this data, but rather immediately in a connected SLC, with a corresponding savings in time.

A second box is shown on the bottom right of Figure 2, in which the above-mentioned commands are listed once again. Next to the commands, there are essential actions, which are performed in the event of these commands in regard to the TLC or SLC starting from possible different starting states of an affected cache memory block. An identical box having identical meaning is also shown under the state transition diagram of Figure 3.

Figure 3 shows a state transition diagram expanded in relation to the state diagram of Figure 2. The state transition diagram shown in Figure 3 has the expanded states SI, ES, MI, and MS. The states II, SS, EM, and MM originate in the sequence indicated from the states I, S, E, and M of the state transition diagram of Figure 2.

The first letter of a state title relates to the state in the TLC, while the second letter relates to the state in the SLC and/or in a lower cache hierarchy stage.

In the exemplary embodiment, instead of the entry EE, the entry EM is selected as the entry for the state description of a cache memory block because, for example, starting from the states SI and/or SS, upon arrival of an RDE command on the part of an individual processor EP at the TLC, the TLC must expect that when an individual processor EP requests a cache memory block Exclusive, it has also modified it. Since the TLC has to maintain the superset via the state situation in the system, it enters the new situation as EM.

In the case that the affected cache memory block is also requested using an RDM command by an individual processor EP, the situation is clear in that the requesting cache memory block is modified. The entry EM in regard to the requested cache memory block is thus the correct result.

Overall, using this method, optimization of the state transition diagram is achieved, because a state situation for which an additional bit position must possibly be provided for its data coding does not have to be shown. Eight states may be coded using three bit positions. In the event of nine and more states, more than three bit positions are necessary.

In the event of an exclusive request (RDE command) of a cache memory block by an individual processor EP, as noted above, the cache memory block is not automatically modified. The requesting individual processor EP has the affected cache memory block Exclusive and may finally perform all possible variations of the handling. For example, it may relay it to other points or displace it. In the first case, the TLC has to have reserved the state ES for the affected cache memory block, in the second case as EI. All of these cases are covered by the TLC using the state description EM, because this provides the certainty that it is maintained in regard to the affected cache memory block in any case when it turns to the affected individual processor EP. The above-mentioned cases are noted in the lower left box of Figure 3 in line five.

Line 5 of the lower left box of Figure 3 corresponds to line 3 in the lower left box of Figure 2. In addition, lines 1, 3, and 8 of the lower left box of Figure 3 correspond, in the sequence indicated, to lines 1, 2, and 4 of the lower left box of Figure 2. The correspondences are each based on the above-mentioned principles.

Since the principles which the state transition diagram of Figure 2 are based on are also used as the basis of the state transition diagram of Figure 3, the description of the state transition diagram of Figure 3 is essentially restricted to the description of the differences. The individual state transitions and the assumption of states when corresponding commands are provided may be inferred from Figure 3.

According to the state transition diagram of Figure 3, there are two Shared states, two Exclusive states, and three Modified states in regard to a TLC. With prefetch requests, for example, it is clear that an entry retrieved in the TLC is only present in the affected TLC. For this situation, the state description SI applies. If this special case, in addition to which there may be others, is marked extra, in the event of requests for this entry, a query on other levels may be dispensed with. Capacity thus becomes free on the side of the first buses BS1, which may be used for other processing by the individual processor EP to increase the performance of the processor system.

The state situation SI is noted in the lower left box of Figure 3 in line 2.

For the exclusive states, there is the additional state ES in regard to the TLC. This state is obtained, for example, when an entry existing Exclusive in the TLC is read shared by an SLC. In this case, the TLC maintains the state E and notes the cache memory block as state S for itself for the SLC. The TLC then knows that the state may only actually be S or I in the SLC. The state may not be M or E, because such an authorization query has not been received. The entry may thus not be changed in any case. The advantage of this knowledge will be explained in greater detail in connection with Figure 4.

The possible states for the SLC level are specified in the lower left box of Figure 3 in line 4.

In addition to the state description MM, the expanded state situations MI and MS are specified in Figure 3.

If an SLC has retrieved an entry of the TLC Exclusive, modified it, and then written it back in the TLC using the WR command, the TLC assumes the state MI for this cache memory block. The SLC has displaced the entry and is no longer valid there. The TLC reaches the same situation when it had a modified entry, i.e., was in the state MM for this entry, and has written on the entry once again using the WR command. The TLC also assumes the state MI then. In this state, the TLC knows that the affected entry may no longer be at an SLC, for example.

The situation is noted in the lower left box of Figure 3 in line 6.

Finally, there is also the expanded state MS, which occurs independently of whether the state I or M is entered for an SLC. The state MS occurs when an SLC reads Shared. As already noted several times above, however, an SLC may cause a logically reduced state after a corresponding action, so that finally one of the states I or S may be behind the state description MS for an affected SLC.

This state of affairs is noted in the lower left box of Figure 3 in line 7.

The advantage of the expanded cache states is that the TLC has a deeper knowledge about the state situation within the processor system, which allows it to relay and/or capture requests directed to the individual processors EP in a more targeted way, for example. The latter may occur because the requests may be handled by the TLC itself. The load of the processor side is thus reduced. The individual processors EP may perceive tasks for increasing the efficiency of the processor system more strongly.

Several cases in which savings in regard to requests on the processor side are possible are shown in Figure 4.

Two external examples have been used as the example. An external command is to be understood, as already noted, as a command which is directed from the system bus side to the TLC. One external command is an RDS command and the other is an RDE command.

The expanded MESI states are shown in the lower part of the figure in Figure 4 and the case of the current four MESI states are shown opposite in an upper part of figure.

If a cache memory block in the TLC has the state I or II, the result is the same in both cases. In both cases, no measures are to be performed. In both cases, there is a MISS case, i.e., no data may be provided.

If the state of the entry affected by the RDS or RDE command is marked in the TLC as S, then no measures are necessary in the event of an RDS command both in a four state model and also in the expanded state model, because the entry is already S in the TLC. In regard to the expanded state model, it makes no difference whether the state SI or SS has actually been provided.

In the event of the RDE command, it is to be ensured both in regard to the TLC and also the SLC in a four-element state model that the state of S assigned to the affected cache memory block is altered to I. The affected cache memory block is then required Exclusive for the side of the individual processors EP. This stresses the processor bus, which is therefore temporarily unavailable for other tasks.

With the expanded state model, another request at the processor side is not always required in regard to the RDE command. For example, if the state noted in regard to the requested entry at the TLC was SI, it is no longer necessary to ensure from the TLC that an affected SLC changes its state notation for the affected cache memory block to the state I. The TLC already knows that the affected cache memory block is classified as I on the part of the individual processors. In this case, the processor side is thus no longer stressed more. Finally, more processor performance is available for other tasks.

If the state of the entry in the TLC affected by the RDS or RDE command is E, it is required in the four state module and RDS command, before it is ensured that an affected cache memory block is assigned a state S in the TLC and in the lower cache hierarchy, that it be checked whether newer data for the affected cache memory block exists in the meantime somewhere in the processor system, since, as already noted, a cache memory block requested as exclusive may be brought later in the event of corresponding further actions by the requesting unit into one of the states M, E, S, or I. The processor side is thus unproductively stressed in this case.

The corresponding is true in the event of the RDE command, with the difference that the SLC units must be caused to identify the affected cache memory block with the state I. The processor side is also unproductively stressed here.

Using the expanded state module, the performance of the processor system may be increased by saving unproductive output. If a cache memory block has been assigned the state ES in the expanded state model and an external RDS command occurs in regard to this cache memory block, further orders on the part of the TLC in the direction of lower cache hierarchies may be dispensed with. It is already known via the SLC that the affected cache memory block is already in the state S there. Because of the fact that the affected cache memory block has been put in the state S in the lower cache hierarchies, it is known that no modification in regard to the affected cache memory block has occurred in the lower cache hierarchies. Therefore, no more current data may be retrieved on the part of the lower cache hierarchies. Therefore, it neither has to be checked whether more current data exists, nor must it be ensured that a change of the state identifier occurs.

Proceeding from the state identification ES in the TLC and an existing external RDE command, it is only to be ensured, since it is known no newer data may exist, as noted above, that the cache memory block is identified by I in the lower cache hierarchies. For this purpose, only a simple request on the part of the lower cache hierarchies is required. Overall, the unproductive output of the processor system is lower than if it must additionally be checked beforehand whether even newer data possibly exists, as in the fourth state model.

The savings potential is particularly clear in the case in which cache memory blocks must be put in the state M in the TLC. In regard to the four state model, in which there is no deeper division of the modified state, it is true for both the RDS command and also for the RDE command that all measures must always be taken in order to check whether even newer data exists, before the affected cache memory block is caused to be identified with the state S or I.

The expanded state model allows a differentiation of the Modified state. In the present exemplary embodiment, the Modified state is divided into the states MI, MS, and MM. If an external RDS command is directed to the TLC, in the cases MI and MS, analogously to the statements above, any request to lower cache hierarchies may be dispensed with. If an RDE command exists, it at least does not have to be checked whether newer data exists. Simple requests in the direction of lower cache hierarchies suffice, through which it is ensured that the state identifier for the affected cache memory block is altered to I in the lower cache hierarchies.

The positions in which measures are required not at all or only restrictedly are identified by an exclamation point in Figure 4. Unproductive stresses of the processor system may be saved in regard to these positions.

PATENT CLAIMS

1. A device for increasing the performance capability of processor systems comprising a large number of individual processors and multiple cache memories, which are designed, to maintain cache coherency among one another, to process a cache protocol operating according to a MESI standard, characterized in that the cache protocol operating according to the MESI standard is configured in such a way that in the indication of MESI states, a multiple indication is at least partially provided in such a way that at least a part of both the MESI state of an affected cache memory (CS) and also the MESI state of an individual processor (EP) connected thereto and/or another cache memory (CS) connected thereto are indicated.

2. The device according to Claim 1, characterized in that the processor system has individual processors (EP), which at least partially have associated first and/or first and second cache memories, and the processor system has additional cache memories (CS), which are situated at least in the first hierarchy stage and to which individual processors (EP) and/or further additional cache memories (CS) are connected individually or in groups, purely or mixed.

3. The device according to Claim 1, characterized in that MESI state combinations such as II, SI, SS, EM, ES, MI, MS, or MM are indicated by the multiple indication.

FIG 1

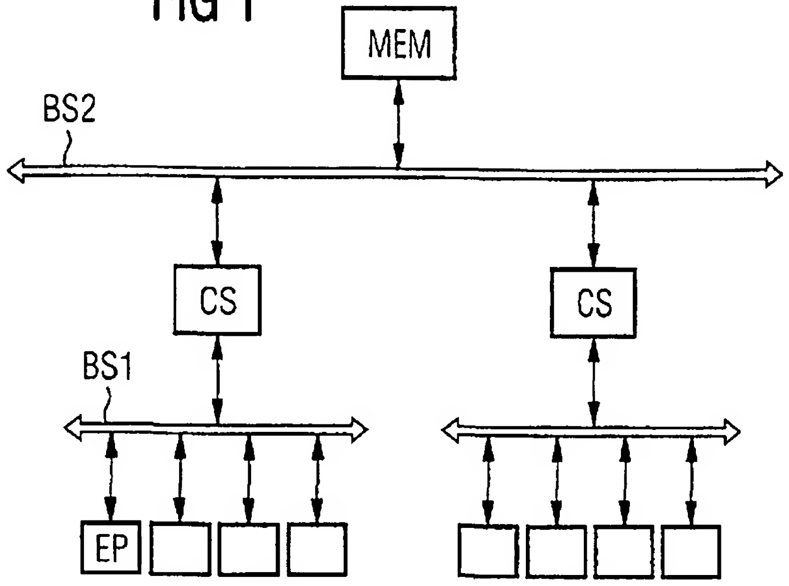
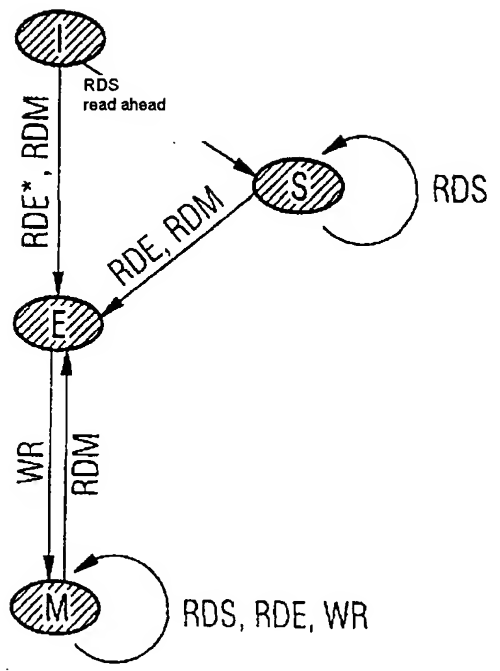


FIG 4

state	external RDS TLC	external RDS SLC	external RDE TLC	external RDE SLC
I	-	-	-	-
S	-	-	invalid	invalid
E	divided	divided	invalid	invalid
M	divided	divided	invalid	invalid
II	-	-	-	-
SI	-	-	invalid	(!)
SS	-	-	invalid	invalid
ES	divided	(!)	invalid	invalid(!)
EM	divided	divided	invalid	invalid
MI	divided	(!)	invalid	(!)
MS	divided	(!)	invalid	invalid(!)
MM	divided	divided	invalid	invalid

BEST AVAILABLE COPY

FIG 2

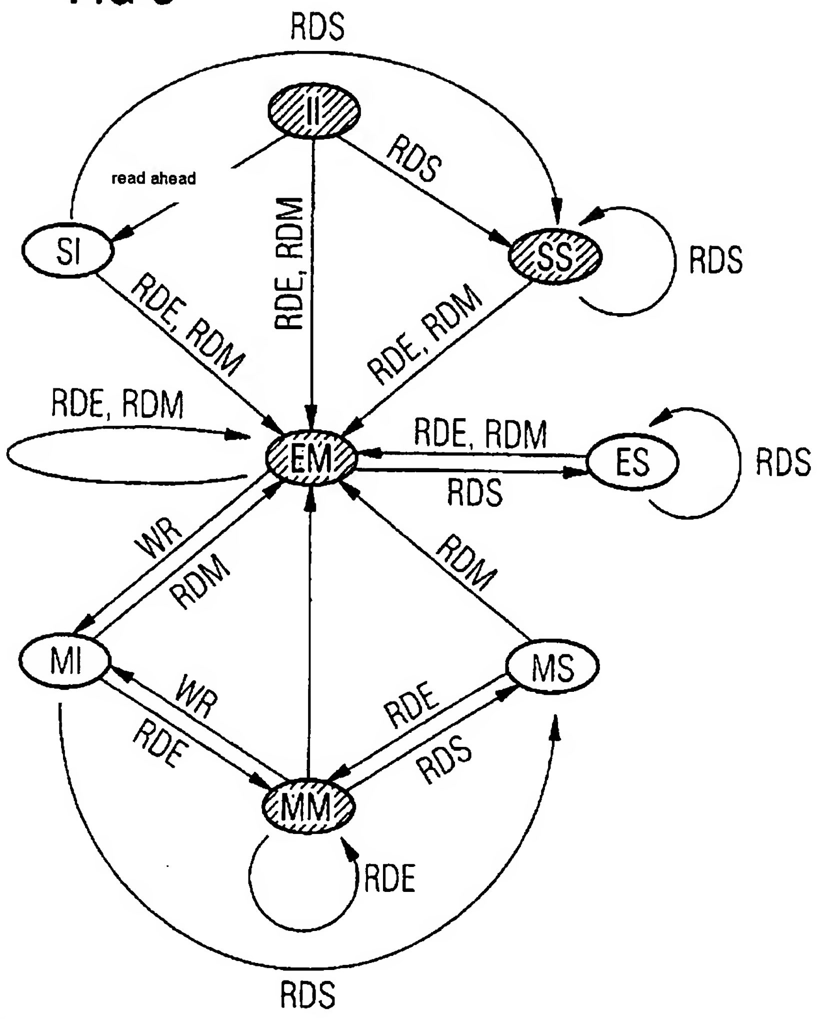


state	TLC state	SLC state
I	I	I
S	S	I, S
E	E	I, S, E, M
M	M	I, S, E, M

command	action
read ahead	TLC: I => S
RDS	SLC: I => S
RDE	SLC: I, S => E
RDM	SLC: I,S => M
WR	SLC: M => I

BEST AVAILABLE COPY

FIG 3



state	TLC state	SLC state
II	I	I
SI	S	I
SS	S	I, S
ES	E	I,S
EM	E	I, S, E, M
MI	M	I
MS	M	I, S
MM	M	I, S, E, M

command	action
read ahead	TLC: I => S
RDS	SLC: I => S
RDE	SLC: I, S => E
RDM	SLC; I,S => M
WR	SLC: M => I
	TLC command required

BEST AVAILABLE COPY